DYNAMICS OF LOCAL GRID MANIPULATIONS FOR INTERNAL FLOW PROBLEMS

Peter R. Eiseman*
Program Development Corporation
White Plains, New York 10601

Aaron Snyder and Yung K. Choo
NASA Lewis Research Center
Cleveland, Ohio 44135

ABSTRACT

The control point method of algebraic grid generation is briefly reviewed. The review proceeds from the general statement of the method in two dimensions unencumbered by detailed mathematical formulation. The method is supported by an introspective discussion which provides the basis for confidence in the approach. The more complex three-dimensional formulation is then presented as a natural generalization. Application of the method is carried out through two-dimensional examples which demonstrate the technique.

INTRODUCTION

The **Control Point Form (CPF)** of algebraic grid generation (reference 1,2) has recently emerged as a powerful interactive tool for a wide range of geometrical applications. While the basic action comes from the motion of a single control point, various automatic features are evolving for a group of control points. The current implementation of **CPF** is being extended from simple two-dimensional geometries to complex three-dimensional configurations. To keep pace with this trend toward increased problem complexity and the implied increase in required grid points, enhancements to the method are being pursued. A primary method of enhancement which is quite fashionable, in light of the current computer technology, is the development of robust automation procedures. This practical avenue is proceeding along several fronts spurred chiefly by the desire to maintain operational manageability. The benefits derived from increased manageability become more pronounced as the geometrical complexity increases. Self-sustaining strategies to deal with the complexity issue, such as establishing a multi-block environment, do not, in general, directly address the issue as viewed from the **CPF** perspective. However, operating within a multi-block surrounding does provide a natural network supporting numerous automation strategies.

* Support given by The U. S. Air Force under AFOSR contract F49620-89-C-0096 and by NASA under Grant NAG 3-877 when author was with Columbia University.

In the current codes, the basic strategy of block interface manipulation has been demonstrated. This is addressed specifically here in the manipulations required to successfully treat periodic boundaries for cascades. This occurs by the periodic tie of two control points and use of free form boundary capability. The potential gain achievable by automating unit operations becomes even more obvious as the number of required points increases.

The actions where it is particularly advantageous to be performed automatically include motion of junctures between blocks, enforcement of boundary orthogonality, and the creation of desired distributions such as uniform or expanding. For example, the objective in implementing automatic boundary orthogonality is the construction of a single command to concurrently reposition all the control points adjacent to a given boundary sector. This would, when chosen, supplant the corresponding tedious interactive option of adjusting each control point successively.

Further features include the dynamic adjustment of the number of control points in each direction and the free form construction of boundaries with more control points than are employed in the given directions within the grid blocks.

Another prominent issue arises when it is necessary to maintain a prescribed boundary geometry while generating a grid for this boundary curve or surface. Enhancements along this line would be particularly welcome and beneficial. Numerical simulations of fluid flow, particularly in the cases where small perturbations in geometry may induce significant changes in the predicted flow field variables, constitute a conspicuous arena where advances in this direction would be highly profitable. Continuing with the subject of fluid flow, the boundary orthogonality procedure alluded to previously can be called upon repeatedly. Through this mechanism, it is possible to extend a simple grid into the interior flow field. This facility requires the automatic movement of bands of control points adjacent to the region of interest.

Since the method is inherently general, the capability exists to advance the automation features to the point where all control points would be determined simultaneously. This brings the stage where solution adaptive techniques could be applied given an arbitrary grid and a reasonable process of attaching a control net to that grid.

OVERVIEW

This section will present the discussion of the following topics: (1) **CPF** methodology and its construction from the transfinite and multisurface techniques; (2) Initialization process which concerns what basic requirements must be established for a control point formulation; and (3) Movement procedures for points once an initial control net is established. The intent of this section is to provide

a suitable background to discuss the formulation without relying on lengthy mathematical development. The delivery of concepts will be to initially present them in the particular and next generalize as, for example, first give the significance of a concept in two-dimensions and then expand to three-dimensions. The intent of this paper is to provide appreciation for CPF and its utility. If all goes well, the reader can begin to apply the method with only a modest need to acquire elsewhere greater detail for his particular application.

The CPF Methodology

In the following development, we will be concerned with transformations from some rectangular cartesian coordinate domain which covers the field of interest to a general curvilinear coordinate system. Alternative coordinate domains exist as candidates but rectangular cartesian coordinates provide the most suitable framework for this discussion. Aside from historical convenience, it is also a short step from a general cartesian coordinate domain to the domain of indices where the spacing between them is unity. This, of course, assumes a finite representation in the form of a grid so that grid point indices are definable. Coming from the other direction, it is now a simple conceptual task to mark off an index array which is now in a one-to-one correspondence with grid points. While this can be done in a number of ways, the size of the domain is translation invariant. The result is a mapping relating a simple indexed system to a more complex physical space coordinate system. It is also permissible to rescale the domain to an arbitrary rectangular region. This is convenient in some instances. For computations, however, the index space is usually preferred. With this utilization of rectangular domains, we advance to the subject of transfinite interpolation.

Transfinite interpolation is covered in detail in the literature (reference 3,4). A brief presentation is given here to establish its connection with **CPF**. Transfinite interpolation was so named for its attribute of matching the interpolated function at an infinite set of points. In this procedure, a function on a region is represented in terms of functions on the region boundaries. To set this in terms of our rectangular domains as discussed above, the function would be specified along the boundaries in the curvilinear coordinate system. By blending corresponding boundaries values, an interpolated value can be calculated for any point in the region. The transfinite procedure, by design, interpolates exactly along each boundary. For grid generation procedures, transfinite interpolation takes on the simple abstract form.

$$X(\xi,\eta) = U \oplus V = U + V - UV \qquad (1)$$

Here **X** is the two dimensional coordinate vector of position and ξ, η are curvilinear coordinates. The interpretation of **U** and **V** are explained easily in terms of our above discussion. The variable **U** can be considered as a univariate interpolation in computational coordinate directions of the values of the position coordinate and derivatives (up to some order **n**) of the position coordinate. This interpolation would be carried out over a range of curves in the curvilinear direction of choice, say ξ. Then, likewise, **V** would be another univariate interpolation with respect to the curvilinear variable η. This second interpolation would, of course, interpolate the position vector and its derivatives through order n, but along curves of constant ξ in this instance. It was pointed out that the transfinite procedure interpolates exactly along the boundary. With this in mind, we now discuss the significance of the product term and its sign in the above equation. We have already accounted for the terms **U** and **V** as being separate univariate interpolations in the respective curvilinear directions of ξ and η. It is also clear that these two separate interpolations must have common nodes corresponding to their intersection. At these nodes the interpolation is, in effect, being counted twice. The product term **UV** is equivalent, in the overall interpolation, to the contribution arising from the set of intersection nodes of the separate univariate interpolation curves. It can be stated then that **UV** is the product projection term resulting from the separate univariate operations being applied consecutively. This product projection term must bear a negative sign to provide proper cancellation in the net interpolation, since these points have contributed once in the first interpolation **U** and again in the second interpolation phase **V**. By supplying the product term **UV**, the interpolation at boundaries formed of sets of constant ξ and η curves reduces to the values specified by either **U** or **V** as appropriate. This is the designed nature of the transfinite interpolation procedure and is most easily visualized in the special case where the interpolation is specified only in terms of the bounding curves. The set of points involving the product term **UV** is then merely the four corner points of the region.

We now wish to state a few characteristics that should not be overlooked. First, the transfinite procedure is a Boolean sum procedure and is well defined when the order of carrying out the operation is immaterial. Second, the product projector term is an important interpolator in its own right, but one which interpolates over the set of intersection points. It is often referred to as the Tensor product interpolator. Third, the extension to higher dimensions is conceptually straightforward. Fourth, many variations exist in the specification of the number and choice of curves as well as in the choice of interpolating functions. In continuation, we consider the three-dimensional environment where both curves and surfaces represent the basic construction elements.

Following the above nomenclature, the transfinite interpolation in three dimensions can be compactly expressed as a sum of interpolation operators. Again, in the context of coordinate generation, it is

appropriate to express this in terms of the general position vector as follows:

$$X(\xi,\eta,\zeta) = U \oplus V \oplus W = U + V + W - UV - UW - VW + UVW \qquad (2)$$

The extra curvilinear coordinate ζ has been introduced as well as the interpolation function **W**. It remains to discuss the significance of the respective terms in the above equation.

To reveal the structure in three-dimensions, it is convenient to take the special case where the interpolation is in terms only of the specified functions on the bounding surfaces. To visualize this, consider a rectangular region with **U**, **V**, and **W** each specified on respective pairs of opposing surfaces. These surfaces comprise the boundary of the rectangular domain. Each pair of opposing surfaces is identified by the variable held separately constant.

With regard to the curvilinear variable ξ, then **U** would be a specified function of η and ζ on one boundary and another function of these two variables on the opposing boundary. This assignment proceeds in a cyclical fashion for the **V** and **W** boundary surfaces which are labeled here by constant η and ζ, respectively. We now examine the origin of each of the terms in the three-dimensional transfinite interpolation expression presented above.

To facilitate this process, let us orient the boundaries such that ξ is directed side-to-side, η top-to-bottom and ζ front-to-back. Then **U** can be the interpolation between side-to-side surfaces, **V** between top-to-bottom surfaces and **W** between front-to-back surfaces. These three interpolations relate to the respective **U**, **V**, **W** terms in the three-dimensional transfinite interpolation equation. The **UV** term arises since the four front-to-back edges formed by the intersection of the side-to-side and top-to-bottom surfaces have each contributed twice, once through **U** and again through **V**. An identical argument holds for the other two sets of four edges corresponding to the product terms **UW**, relating to the top-to-bottom edges; and **VW** relating to the side-to-side edges. Of course, each is fixed with a negative operator. Thus, the three binary product terms are directly related to the three corresponding set of edges as just identified. It now remains to explain the ternary product **UVW**. This is explained by examining the corner points of the region. It can be seen that each corner point contributes once through each of the **U**, **V**, and **W** interpolations and then is subtracted for each of the binary product terms **UV**, **UW**, and **VW**. The effect, so far, is that the eight corner points have not contributed to the overall interpolation. The **UVW** term contains the collective contribution of these points to the interpolation and hence is positive. This establishes the framework to progress to the multi-surface concept and its role in the **CPF** methodology.

We now present a brief description of the multisurface transformation. The geometrical picture in this instance parallels that of the transfinite interpolation procedure laid out above. We will reconstruct the geometry framework so that it will be clear what the connection is between the two procedures.

The multisurface transformation can be viewed as a general technique of connecting coordinates between specified boundaries. In two dimensions, these boundaries can be viewed as distinct curves in space separated by some value of a given curvilinear variable. A second curvilinear variable would vary along these boundary curves. As in the case of transfinite interpolation, the picture easily generalizes to surfaces separated by a constant value of a curvilinear variable. In this instance, it follows that each surface is coincident with a pair of second and third curvilinear variables. Within this framework of an interacting net of families of coordinate lines, the multisurface transformation proceeds by providing for the arbitrary specification of additional curves or surfaces to be used as auxiliary instruments to control grid structure. The discussion will continue with the focus on auxiliary curves with the understanding that this line of argument extends directly to surfaces.

Once a set of auxiliary curves is established, a vector field of smooth tangents is constructed in correspondence with lines connecting the selected auxiliary curves and boundaries. The lines used to construct these tangents are simply defined as vector differences between positions on successive auxiliary curves. The position vector serves as the direct connection between the transfinite and multisurface transformations. The critical differentiation between the two is that, unlike the aforementioned property of the transfinite procedure, the multisurface transformation does not require that the interpolation match any of the auxiliary curves. In the latter case, it is only required that the interpolation match the inner and outer boundaries. In this respect then, the multisurface method described by Eiseman is a very flexible univariate scheme which is similar to Be'zier and B-spline approximation (reference 5), where parameters defining a curve are not necessarily on the curve. The idea of curves extends also to surfaces.

We are now at a point where we can assemble the **CPF** structure from the transfinite interpolation and multisurface transformation. The result will be a procedure whereby a sparse collection of control points along with the specified boundaries is used to form the transformation and then to generate the grid with any number of desired points. The following discussion will indicate how the transfinite and multisurface concepts lead to a class of coordinate transformations, whereby the interior form of the coordinates can be manipulated in a local fashion and whereby any boundary can be specified or manipulated in a similar fashion.

At the outset, we are given a logically ordered array of control points together with specified boundaries. Because of the intrinsic

property of the multisurface constructs, the control points become direct controls over the curvature of the generated curve. The philosophy behind the origin of the net of control points is discussed in the separate initialization section which follows this section.

The specific boundaries are included in the transfinite rather than the tensor product assembly of directions. Recall that the tensor product was introduced above as an interpolation procedure involving a collection of points.

The specific collection of points were the result of either an intersecting net of curves or surfaces. In the present assembly, new constructive elements are introduced by altering the basic parts upon which the multisurface transformation is applied. When the parts are one- or two-dimensional, the first and last elements of each sequence for a multisurface construct are replaced by the specified boundary parts at the corresponding location. This replacement is just a substitution of a specified boundary part for a corresponding part generated by control points. We can review this process by drawing upon our image of the rectangular domain. The assembly proceeds in two dimensions by first taking the sum of the constructs with specified boundaries; by then observing that the sum of a control point curve and a specified curve appears over each domain boundary; and finally, by noting that the specified boundaries can be matched by subtracting the tensor product transformation so that the resultant transformation will match all the boundaries. This represents the Boolean sum process and this assembly is thus transfinite. With some algebraic manipulation, the transfinite form just obtained can be nicely separated into a Tensor product core transformation with four adjustment terms for the boundary blending action. This separation is achieved through the inclusion of on-off factors to switch between specified (transfinite) or free-form (control point) boundaries as the switch values go between one and zero, respectively. In a similar but more complex fashion, this strategy can be established in three-dimensions. In that context, there are now two nontrivial boundary parts represented by faces and edges which can similarly be given on-off switches.

Initialization

Now that the construction of **CPF** has been laid out, it is appropriate to discuss in more detail the initialization procedure. The basic requirements to establish a control point transformation are the specified boundaries, their respective on-off switches, the number of grid points for each direction, the number of control points for each direction and the position of those control points. The last concern is the primary concern since the others are usually introduced as input. Impetus from the desire to be able to automate the process of control point determination and, hence, be free of the tedious manual control is the main guiding force in the following rationale.

The basic automatic control net determination comes from the "attachment" to an existing transformation. The process starts with the existing transformation either in analytical form or in the discrete form of a grid. This allows for considerable flexibility while introducing only a modest constraint. A simple way of giving an analytical form of attachment and perhaps the most efficient is to employ a transfinite interpolation which assembles linear interpolations between all specified opposing boundaries. Of course, as follows from our discussion above, it is a mapping which conforms to all boundaries. Implied mappings exist in the case where a coordinate grid is given. In this instance, it is convenient to index the grid points and thus establish our aforementioned rectangular domain determined by the minimum and maximum index in each direction. As before, the index grid is simply cartesian with unit spacing in each direction. In either the analytical or the discrete case, the common element is the mapping from a given simple domain to the more complicated region of interest. This should sound familiar and it should be of no surprise that in both cases the domain can be assumed to be rectangular. Because of their simplicity, control points can be easily placed in locations that would produce a control point transformation of the rectangle onto itself which would also move no point: that is, we can exactly reproduce the identity map! With this exactness, here, we then consider the more general given map and use it to send the control points from the rectangular domain into the image region of interest. This then automatically defines the control net on the physical region and provides an approximation to the originally given transformation. It turns out in most cases that even with a modest number of points the degree of approximation is quite good. Aside from providing for automatic initialization, we also receive the capability to locally modify virtually any existing transformation be it analytically defined or defined in the form of a grid.

The main part of the initialization concerns the placement of control points in the field. There is also an avenue to specify the boundaries in a control point form with a different number of points. Examine the situation of an increase in the number of control points. For each boundary, the strategy is to attach, as above, to a simply constructed or given boundary and then to move the richer supply of control points about to model the shape into the desired form. The results so obtained can be used either directly or in reparameterized form. In the latter case, the attachment and manipulation process can be employed again, but now in the parameter space.

Movement of Control Points

Once an initial control net is established, a number of strategies exist regarding the movement of these points. Movement can be accomplished through global strategies such as those based on PDE's or more local strategies. In either case, there exists the implicit

advantage of dealing with a relatively sparse set of control points rather than a dense set of grid points. Regardless of the choice of movement method, it can also be viewed as an extension of the initialization process.

Several options exist for the use of local controls which can be conducted efficiently in an interactive environment. This set of options includes the free-form modeling of chosen boundaries, the establishment of orthogonality at segments of boundaries or their sum, the inward propagation of such orthogonality, the creation of local grid clusters, and the local embedding of specific coordinate forms. These actions can be executed on a point by point basis or for a section of points. In the event a section of points is to be moved, a single point is usually moved and in response a neighborhood of points similarly move but in a progressive sense. This process is called rubber banding if one direction is involved. The extension of this process is called rubber sheeting. An exception to this type of collective motion is the establishment of local orthogonality where adjacent points along a boundary must be moved as a unit to maintain derivative continuity. A boundary shared by two distinct coordinate systems would require this manner of uniform control. This boundary situation arises naturally in the use of multi-block procedures. The required action to satisfy orthogonality at boundary junctures can clearly be done singly or collectively in the sense of rubber banding or sheeting.

FORMULATION

We are now in a position to present the explicit formulation of **CPF**, the framework being previously outlined in the methodology section. We first present the **CPF** in its two-dimensional version where the elected variables will be ξ and η. In effect, we specialize to a face of our rectangular region which we can take to be either the face $\zeta = \zeta_1$ or $\zeta = \zeta_N$ which bound the domain in that direction. This removes the ζ variable from the general expression and we give the explicit two-dimensional form as

$$\begin{aligned}
\mathbf{X}(\xi,\eta) &= \mathbf{U}(\xi,\eta) + \mathbf{V}(\xi,\eta) - (\mathbf{UV})(\xi,\eta) \\
&= \mathbf{T}(\xi,\eta) + \rho_1\alpha_1(\xi)[\mathbf{X}(\xi_1,\eta) - \mathbf{A}_1(\eta) \\
&\quad + \rho_2\alpha_{L+1}(\xi)[\mathbf{X}(\xi_L,\eta) - \mathbf{A}_{L+1}(\eta)] \\
&\quad + \rho_3\beta_1(\eta)[\mathbf{X}(\xi,\eta_1) - \mathbf{B}_1(\xi)] \\
&\quad + \rho_4\beta_{M+1}(\eta)[\mathbf{X}(\xi,\eta_M) - \mathbf{B}_{M+1}(\xi)]. \qquad (3)
\end{aligned}$$

Here **X** is our position vector and **U** and **V** are the univariate

interpolation surfaces in the respective curvilinear directions of ξ and ζ, respectively. We define the remaining quantities as we proceed to explain their significance. The boundary edge curves of the selected face formed from the control point attachment process have been labeled by the minimum and maximum subscript index in each direction and appear as $\mathbf{A}_1$ and $\mathbf{A}_{L+1}$ for the curves of constant ξ and $\mathbf{B}_1$, $\mathbf{B}_{M+1}$ for the curves of constant ζ. The weighting from the continuous distribution of these control point curves is thus evidenced through the last term from each of the four bracketed pairs in this equation. In a similar fashion, the first term in each of these brackets represents the specified boundaries. These specified boundaries are identifiable through the corresponding subscript attached to that coordinate held constant. We now examine the four pairs of coefficients premultiplying each of the bracketed terms. The first factor ρ_i represents the on-off switch which is applied independently to its corresponding boundary curve. With the switches all set to off, the interpolation reduces to the tensor product $\mathbf{T}$ consisting solely of the contribution of the control points in the surface. With each additional switch turned on, the grid is restructured by the control features enabled by that switch. The control mechanism is incorporated essentially in the designed difference between the geometry of the specified curves and that represented by the control point curves. The premultiplying factors yet to be discussed are the quantities α and β which smoothly pass control between the opposing boundaries corresponding to their indicated arguments and subscripts. Taken together, the controls available are then primarily used for boundary conformity and for shaping curves connecting opposing boundaries.

The two-dimensional control point formulation just outlined provides for flexible grid control for arbitrary surfaces. The on-off switches enable adjustments to be made along boundary faces independently. The method provides for a situation where one boundary face is specified and the opposing face is left open for manipulation. Adjustments to this second surface can be made through any of the methods previously discussed.

To supply further support that this method of contrived regulation can be established, it is helpful to quickly review the basic strategy underlying the control point curves in two dimensions. The strategy followed is to construct a transfinite transformation matching all the boundaries. This transfinite transformation can be constructed from the sum along each boundary of a specified curve and a control point curve minus the tensor product transformation used to establish the control point curves. Algebraic manipulation is then used to cast this transfinite transformation into a form where a separate tensor product core is manifested. The remaining part is the controlled adjustment terms representing the deviation from a pure control point representation to one of an exact boundary specification. To make this work out in two dimensions, the corner points of the face are properly accounted for in the adjustment terms in order that they will not contribute twice.

Extending the situation to the case of a three-dimensional rectangular region, we can quickly exploit our understanding of the Boolean sum process provided in the methodology section. Recall that the overall process is to be a Boolean sum which reduces to a tensor product core represented by T along with simple adjustment terms for each face or edge of the grid block. Each adjustment term appears as a blending function times the difference between the specified boundary part and the corresponding control point representation for the same part. Considering a given boundary face, the boundary edge blending terms can be separated from the pure control point dependency by properly accounting for the relevant edges. For the UV tensor product in ξ and η, those edges are the four cube edges varying in ζ. We immediately observe that the remaining faces are treated similarly and the relevant four edges are the edges transverse to the face associated with that product. The end result is a process whereby when a given adjustment term is switched off, the corresponding pair of faces and four edges are dropped in combination. In this way, only those control points effecting a given boundary need be considered when manipulating that boundary. The practical implication is that any combination of specified and free formable boundaries can be employed.

We now present the construction of the three-dimensional **CPF**. This construction is a generalization of the two-dimensional form and consequently new as well as more complex elements must be defined. The three-dimensional construction of the **CPF** can be presented in a step-by-step fashion.

First, define $\mathbf{q}_{ijk}$ as a sparse array of control points with index subscripts as such:

$$\{\mathbf{q}_{ijk} : i=1,2,\ldots,L+1,\ j=1,2,\ldots,M+1,\ k=1,2,\ldots,N+1\}, \qquad (4)$$

which establishes three sets of control point sequences whose associated end conditions are (i = 1 or L+1, j = 1 or M+1, k = 1 or N+1) for the three directions ξ, η, and ζ, respectively. It is clear that any normal sequence can be used to generate a curve which connects the first and last control points of the sequence.

Second, define curves connecting the first and last control points. This is done for each of the three families of curves by blending the corresponding control points for each sequence using the summation as follows:

$$\mathbf{a}_{jk}(\xi) = \sum_{i=1}^{L+1} \alpha_i(\xi)\, \mathbf{q}_{ijk}, \qquad (5a)$$

$$b_{ik}(\eta) = \sum_{j=1}^{M+1} \beta_j(\eta)\, q_{ijk}, \tag{5b}$$

$$c_{ij}(\zeta) = \sum_{k=1}^{N+1} \gamma_k(\zeta)\, q_{ijk}, \tag{5c}$$

where α and β are the blending functions introduced previously for the two-dimensional construction and γ is the additional function required here.

Third, The curves just defined are used to construct surfaces which match the control points at their corners. The surfaces are generated by three sets of double summations which are given as the following single indexed coefficients:

$$A_i(\eta,\zeta) = \sum_{j=1}^{M+1}\sum_{k=1}^{N+1} \beta_j(\eta)\gamma_k(\zeta)\, q_{ijk}, \tag{6a}$$

$$B_j(\xi,\zeta) = \sum_{i=1}^{L+1}\sum_{k=1}^{N+1} \alpha_i(\xi)\gamma_k(\zeta)\, q_{ijk}, \tag{6b}$$

$$C_k(\xi,\eta) = \sum_{i=1}^{L+1}\sum_{j=1}^{M+1} \alpha_i(\xi)\beta_j(\eta)\, q_{ijk}, \tag{6c}$$

Fourth, the coarse array of control points are used to construct the tensor product core. This is done simply in terms of the blending functions used for the surface construction and is given by

$$T(\xi,\eta,\zeta) = \sum_{i=1}^{L+1}\sum_{j=1}^{M+1}\sum_{k=1}^{N+1} \alpha_i(\xi)\beta_j(\eta)\gamma_k(\zeta)\, q_{ijk}. \tag{7}$$

This tensor product core corresponds to the **UVW** term in our three-dimensional Boolean expression for the general position vector $\mathbf{X}(\xi,\eta,\zeta)$.

Fifth, we construct the adjustment terms corresponding to the surfaces and edges of the array boundary. The adjustment terms are a

generalization of the two-dimensional construction. It can be presented in two steps. Each step is formed around the tensor product core. Thus, for the first adustment associated with the surfaces, the univariate interpolations are given by the following:

$$U(\xi,\eta,\zeta) = T(\xi,\eta,\zeta) + \alpha_1(\xi)[X(\xi_1,\eta,\zeta) - A_1(\eta,\zeta)] + \alpha_{L+1}(\xi)[X(\xi_L,\eta,\zeta) - A_{L+1}(\eta,\zeta)], \quad (8a)$$

$$V(\xi,\eta,\zeta) = T(\xi,\eta,\zeta) + \beta_1(\eta)[X(\xi,\eta_1,\zeta) - B_1(\xi,\zeta)] + \beta_{M+1}(\eta)[X(\xi,\eta_M,\zeta) - B_{M+1}(\xi,\zeta)], \quad (8b)$$

$$W(\xi,\eta,\zeta) = T(\xi,\eta,\zeta) + \gamma_1(\zeta)[X(\xi,\eta,\zeta_1) - C_1(\xi,\eta)] + \gamma_{N+1}(\zeta)[X(\xi,\eta,\zeta_N) - C_{N+1}(\xi,\eta)]. \quad (8c)$$

This establishes the adjustment terms for the boundary surfaces. Notice that the tensor product core appears in each univariate construction. These correspond to the **U**, **V**, and **W** terms in the Boolean expression for the general position vector. To complete the process, the second step deals with the three product terms **UV**, **UW**, and **VW** of the Boolean sum. Again by design, each of these three products are constructed around the tensor product core. To eliminate redundancy, only the **UV** product construction is written out. The remaining two can be achieved by cyclic substitution. This product term takes the following:

$$UV = T(\xi,\eta,\zeta) +$$

$$\alpha_1(\xi)\beta_1(\eta)\{X(\xi_1,\eta_1,\zeta) - c_{1,1}(\zeta)\} + \alpha_1(\xi)\beta_{M+1}(\eta)\{X(\xi_1,\eta_M,\zeta) - c_{1,M+1}(\zeta)\}$$

$$+ \alpha_{L+1}(\xi)\beta_1(\eta)\{X(\xi_L,\eta_1,\zeta) - c_{L+1,1}(\zeta)\} + \alpha_{L+1}(\xi)\beta_{M+1}(\eta)\{X(\xi_L,\eta_M,\zeta) - c_{L+1,M+1}(\zeta)\}. \quad (9)$$

Since each of the three binary product terms appear as negative terms in the Boolean sum, the three tensor product terms arising here collectively cancel those three arising in the **U**, **V**, and **W** constuctions. We are thus left with the single tensor product core due to the **UVW** term and the three pairs of surface adjustments and finally the three sets of corresponding edge adjustments. This completes the construction process.

INTERACTIVE GRID GENERATION EXAMPLES

Figure 1 shows an example of a two-dimensional control point array ($\mathbf{q}_{ij}$). A fundamental part of the **CPF** is the construction of coordinate curves, whose shape and location are controlled by the control points. Construction of a coordinate curve, $\mathbf{E}_2(r)$, is illustrated in the figure. The basic interactive process of generating grids using the **CPF** is illustrated in figure 2 using the basic computer program called CPGRID (reference 1). It starts with the construction of a control net using transfinite interpolation and

a surface grid. An initial grid is generated and examined. If desired, the grid can be improved by changing the structure of the control net. In this example a control point is moved to a new position to obtain fine meshes in the middle of the flow domain.

A family of menu-driven interactive grid generation programs (TurboI and TurboT) is being developed using the **CPF**. Several features of the programs allow the global control net to be conveniently changed by the user. The point by point modification of the control net is then used to make a more precise local change. TurboI (reference 6) generates grids for flow simulations in internal flow passages such as inlets, nozzles, and ducts. TurboT (reference 7) is being tailored for turbomachinery. Both programs run on IRIS 4D workstations.

Figure 3 shows an H-grid generated by using TurboI for a turbine stator vane. In this example, the user used only those basic interactive features (i.e., the point by point control of the control net) which are illustrated in figure 2. Figure 3(a) shows an initial control net; figure 3(b) shows an initial grid. The modified control net and grid are shown in parts (c) and (d) of the figure, respectively. The modified grid is nearly orthogonal to the vane surface and is densely clustered around the leading and trailing edges.

The dynamic nature of the interactive process of TurboI is illustrated in figures 4 through 8 using a grid generation example for a converging/diverging axisymmetric nozzle. After logging on an IRIS 4D workstation, the user should type in TurboI to begin the interactive process of the program. The system then asks for the input filename. When the user types in the input filename, TurboI reads in the data, and displays it as shown in figure 4 with menus. To continue the process, the user selects the option called "RESUME". The workstation mouse is all that is needed to select a menu option. Reference 2 explains about the program TurboI and its interactive features in detail.

Selection of the menu option "RESUME" prompts TurboI to construct initial control net by using a simple linear interpolation. TurboI then proceeds to generate the field grid using the control point form, and displays both the grid and control net on the screen. The "SCROLL VIEW" option allows the user to examine either the grid alone, the control net alone, or the grid and control net together. To examine the grid more closely about a region of interest, the menu option "ZOOM & MOVE" is available. The initial control net and initial grid are shown in parts (a) and (b) of figure 5, respectively. In this example, the initial grid shows that it has a slope discontinuity across the symmetry axis and non-orthogonal grid along the nozzle surface.

For illustration purpose, suppose our objectives of this exercise are (a) to make the grid orthogonal to the nozzle surface and (b) to

have slope continuity across the horizontal symmetry axis.

To accomplish the above objectives, the option called "MODIFY CONTROL NET" should be chosen. Then, a set of menus appears on the screen. Selecting the option called "Normalize at TOP (2 level)" is the first step to make the grid orthogonal to the nozzle surface. In order to obtain slope continuity across the symmetry axis, a menu option called "Normalize Bottom (1 level)" is selected. The modified control net at this stage is shown in figure 6(a). Selection of the menu called "Recalculate GRID" will then recompute the grid based on the modified control net. This modified grid shown in figure 6(b), however, does not have slope continuity across the symmetry axis in spite of the orthogonal control net. This is because the grid points on the symmetry axis are not allowed to change (Dirichlet boundary condition is enforced). To modify the grid on the symmetry axis, choose the CN line of that boundary by choosing the menu item "Move j-Net" until the desired boundary CN curve is highlighted in red. Then the menu item called "Free Form Boundary" should be selected to allow movement of grid points along the boundary to conform the orthogonal control net (Neuman boundary condition.) Choosing the option "FREE FORM Boundary" is equivalent to turning off one of the boundary on-off switches (ρ_i) of equation (3). Selecting the option "Recalculate GRID" produces the new grid shown in figure 6(c). This grid meets the objective of this example.

Further improvement (or fine tuning) of mesh structure can easily be done with TurboI. Examination of the internal mesh of figure 6(c) shows that changes in the slope of some coordinate curves are larger than necessary. To cut down on these large changes, a mesh control feature called "SNAP" is used. This allows for smoothing of a CN curve as if a RUBBER BAND were stretched between the current CN point at q_{34} and a chosen "Hinge Point" in a circular mark at q_{32} in figure 7(a). On the workstation screen, the current control point is an intersection of two highlighted control-net lines. Selecting the menu option called "SNAP" changes the CN line segment between the hinge point and the current CN point to the one shown in figure 7(b). A user can then move on to the next CN curve and "SNAP" it if desired. At the end of this fine tuning process, the control net has the structure as shown in figure 7(c). Finally, select the option "Recalculate GRID". The new grid calculated from the fine tuned control net is shown in figure 8(a). It is zoomed and shown in figure 8(b).

Many grid generation programs do not allow local mesh control as TurboT does. For TurboT the initial grid may be generated by any program familiar to a user; then an initial control net may be obtained from the grid by attachment to produce a grid structure that is essentially similar to the initial one. The initial control net shown in figure 9(a) was constructed by attachment. Figure 10(a) shows an initial grid of a compressor rotor blade. Once the control net is created, the interactive process to be followed is very similar to the process of TurboI. The control net is modified to the one

shown in figure 9(b), and a new grid shown in figure 10(b) is generated from the modified control net. The modified grid is more orthogonal and has slope continuity across the periodic boundary. The shape of the control net can easily be changed by using an interactive process illustrated in figure 11. In part (a) of the figure, a user first chooses a control line to be changed and then picks a hinge point. In part (b), control point 1 is moved to point 2 by moving the mouse of the workstation. Point 1' automatically moves to point 2' while maintaining a pitch with the points 1 and 2, respectively. In part (c), the desired shape of the control line is obtained by choosing a menu option called "SNAP" which makes the control line stretch like a rubberband.

CONCLUSION

Our objective in this work has been to report on a simply structured scheme of algebraic grid generation. This scheme is called **CPF** and a number of grid generation codes now employ it. Several underlying features of **CPF** make it an ideal candidate for existing and emerging applications. These features include the following: the capability of conforming to boundaries; the quality of being easily manipulated by numerous local and global grid distribution strategies, concise structure of its formulation which enables its straightforward implementation, and the inherent quality of being compatible with various complementary or supportive operations. Many of these beneficial features are already being exploited. Their usefulness has been demonstrated here by way of two-dimensional examples illustrating basic interactive features. These examples pertain to flow simulations specific to turbomachinery and internal flow passages such as inlets, nozzles, and ducts. Notwithstanding the restricted range of application considered here, **CPF** is in fact a general procedure with wide-ranging potential for application.

REFERENCES

1. Eiseman, P.R., "A Control Point Form of Algebraic Grid Generation," Intl. J. Numerical Methods in Fluids, Vol. 8, pp. 1165-1181, 1988.

2. Eiseman, P.R., Choo, Y.K., Smith, R.E., "Algebraic Grid Generation with Control Points," Finite Elements in Fluids, Vol. 8, edited by T. J. Chung, Hemisphere Publishing Corporation, to appear in 1990.

3. Gordon, W.J., Thiel, L.C., "Transfinite Mappings and their Application to Grid Generation," Numerical Grid Generation, Edited by J. F. Thompson, North-Holland, 1982, pp. 171-192.

4. Smith, R.E., Eriksson, L.E., "Algebraic Grid Generation," Computer

Methods in Applied Mechanics and Engineering, vol. 64, no. 1-3, Oct. 1987, pp. 285-300.

5. Eiseman, P.R., "Coordinate Generation with Precise Control over Mesh Properties," J. Comp. Phys., Vol. 47, No.3, pp. 331-351, 1982.

6. Choo, Y.K., Reno C., Van Overbeke T., Sosoka, D., "Interactive Grid Generation Using Control Point Form, "Program TurboI for Inlets, Nozzles, and Ducts, version 01.01, to be printed as a NASA TM.

7. Choo, Y.K., Soh, W., Yoon, S., "Application of a Lower-Upper Implicit Scheme and an Interactive Grid Generation for Turbomachinery Flow Field Simulations," ASME Paper 89-GT-20, June 1989.

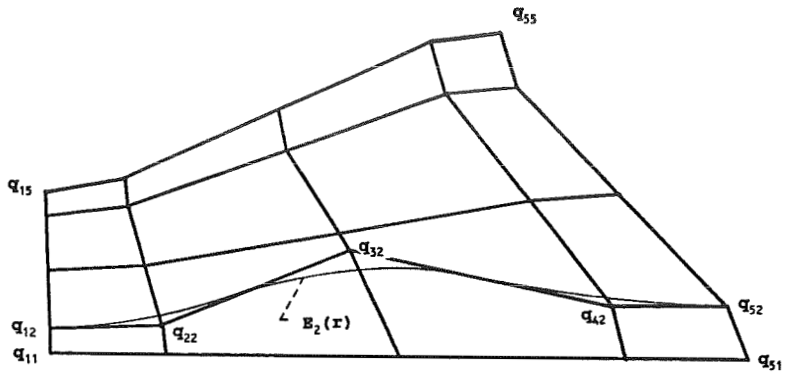


FIGURE 1. CONTROL NET - AN EXAMPLE IN TWO DIMENSIONS

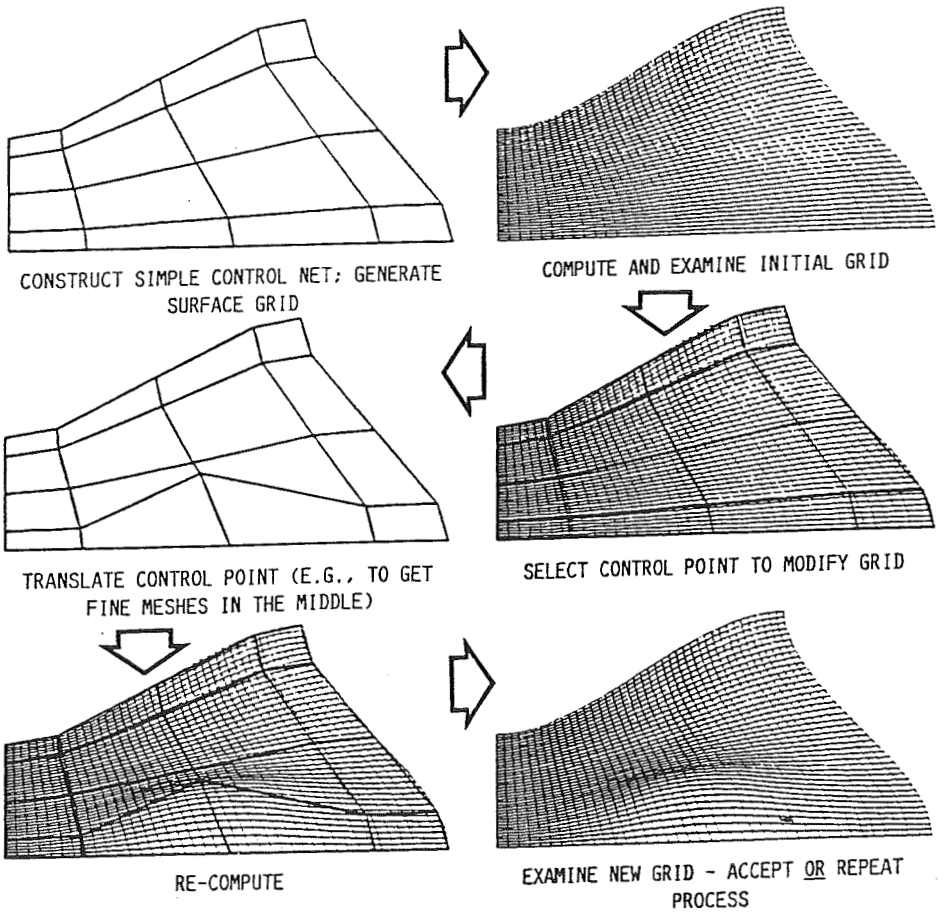
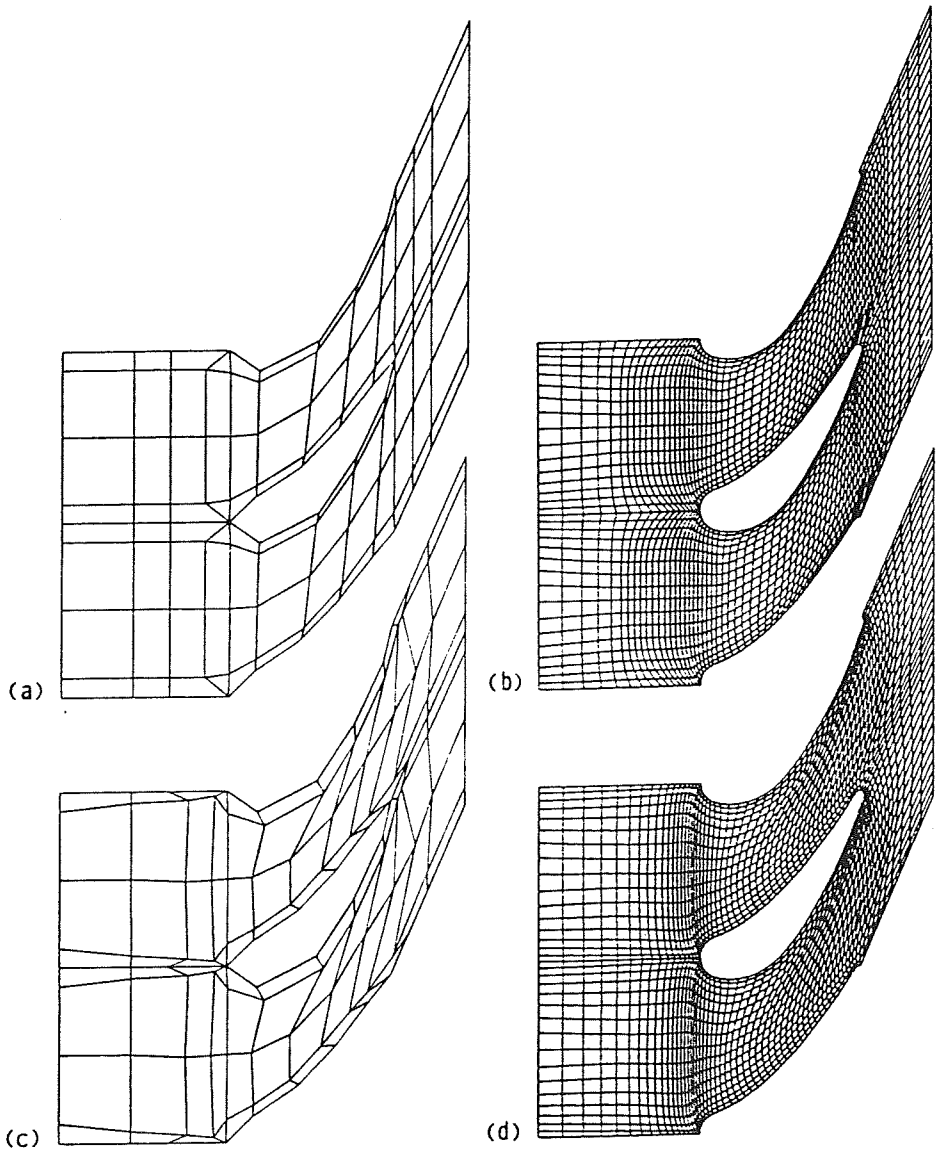


FIGURE 2. BASIC INTERACTIVE PROCESS



(a) Initial control net. (b) Initial grid.
(c) Modified control net. (d) Modified grid.

FIGURE 3. GRID GENERATION FOR A STATOR VANE

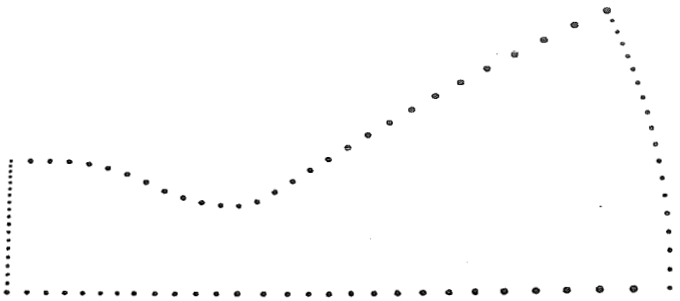
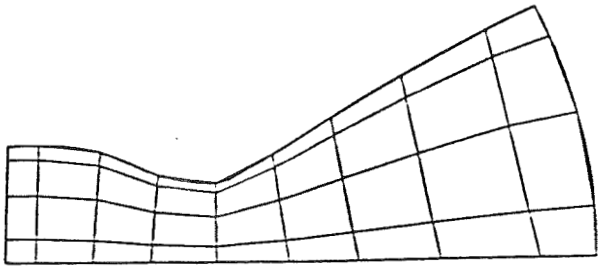
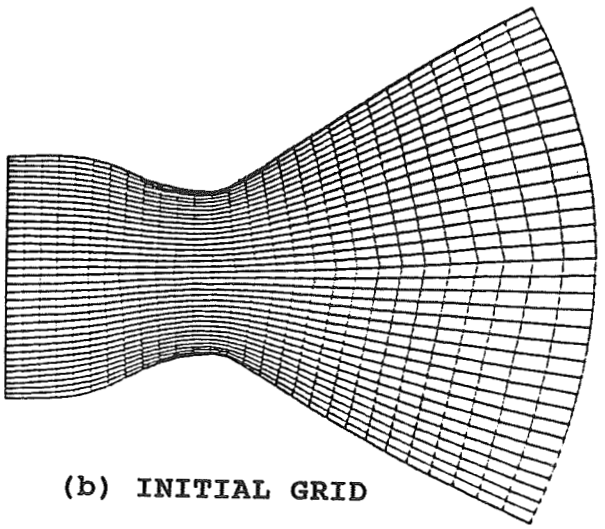


FIGURE 4. BOUNDARY GRID OF A CONVERGING/DIVERGING NOZZLE

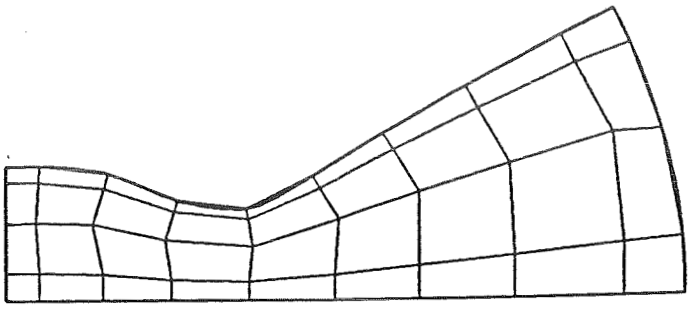


(a) INITIAL CONTROL NET

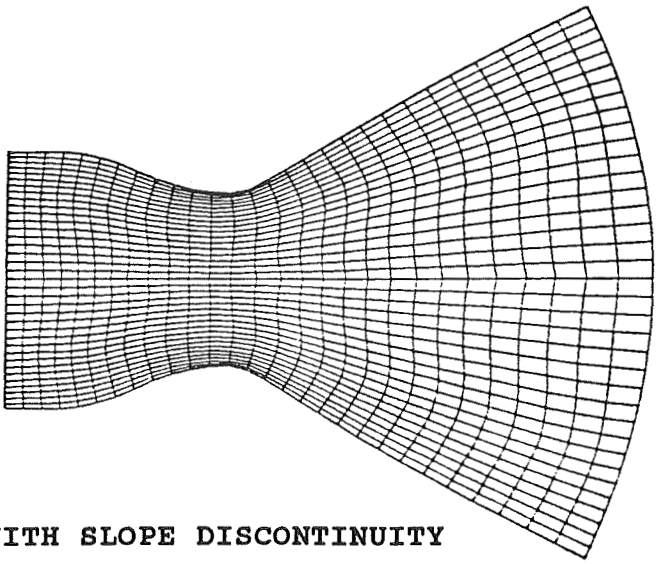


(b) INITIAL GRID

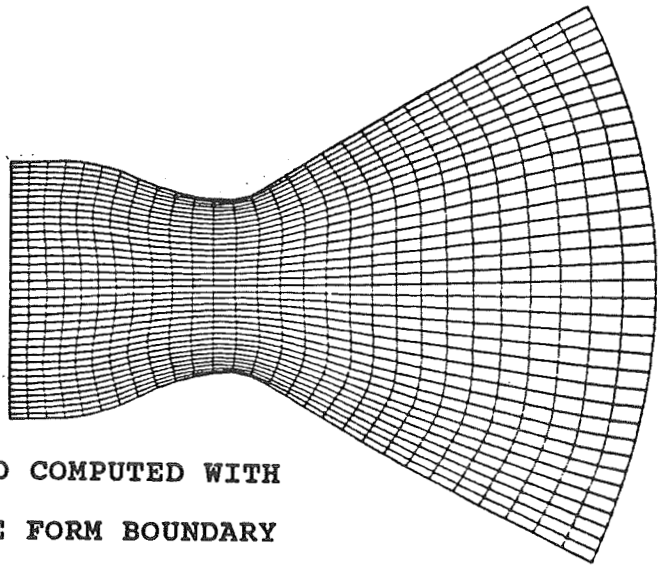
FIGURE 5. INITIAL RESULTS



(a) CONTROL NET ORTHOGONAL TO TOP AND BOTTOM BOUNDARIES

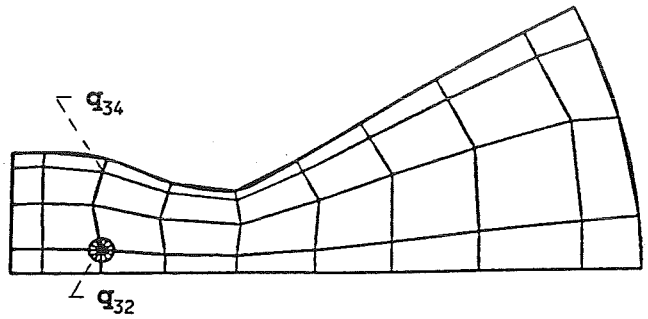


(b) GRID WITH SLOPE DISCONTINUITY ACROSS THE SYMMETRY AXIS

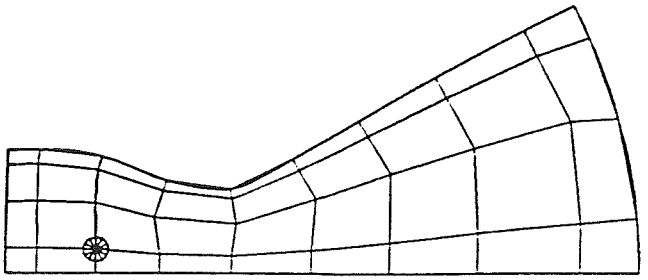


(c) GRID COMPUTED WITH FREE FORM BOUNDARY

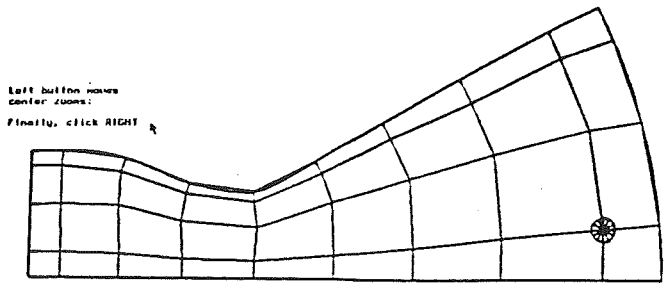
FIGURE 6. INTERACTIVE PROCESS OF GRID MODIFICATION



(a) BEFORE THE "SNAP"

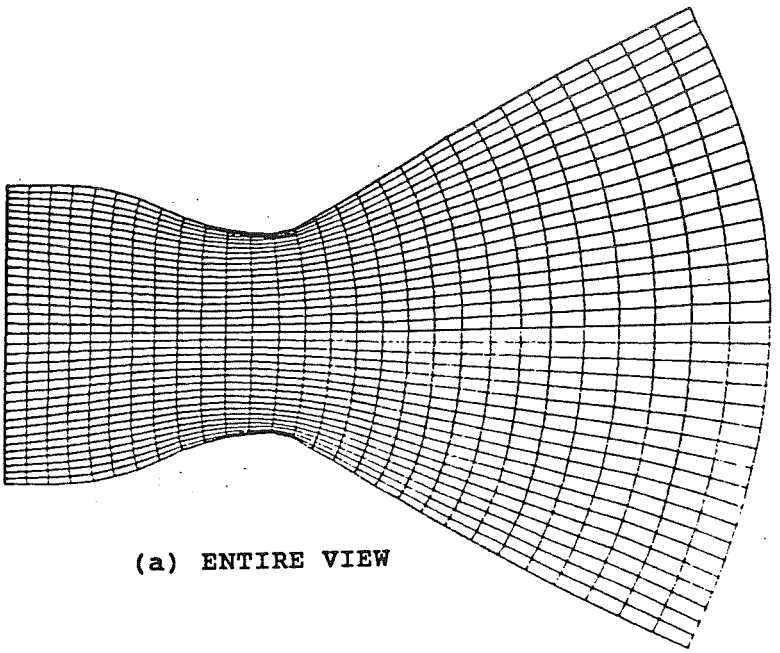


(b) AFTER THE "SNAP"

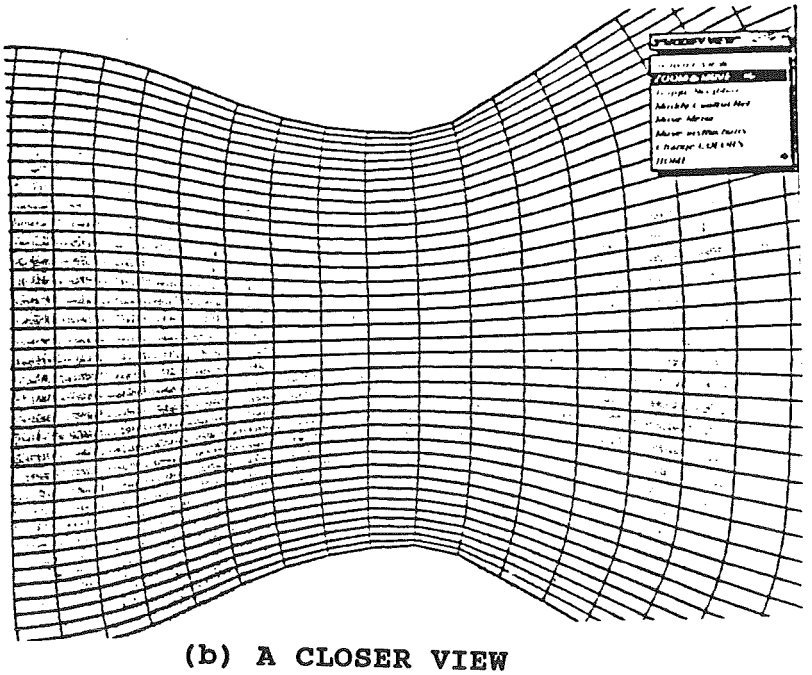


(c) END OF FINE TUNING

FIGURE 7. INTERACTIVE PROCESS FOR FURTHER MODIFICATION



(a) ENTIRE VIEW



(b) A CLOSER VIEW

FIGURE 8. FINAL MODIFIED GRID

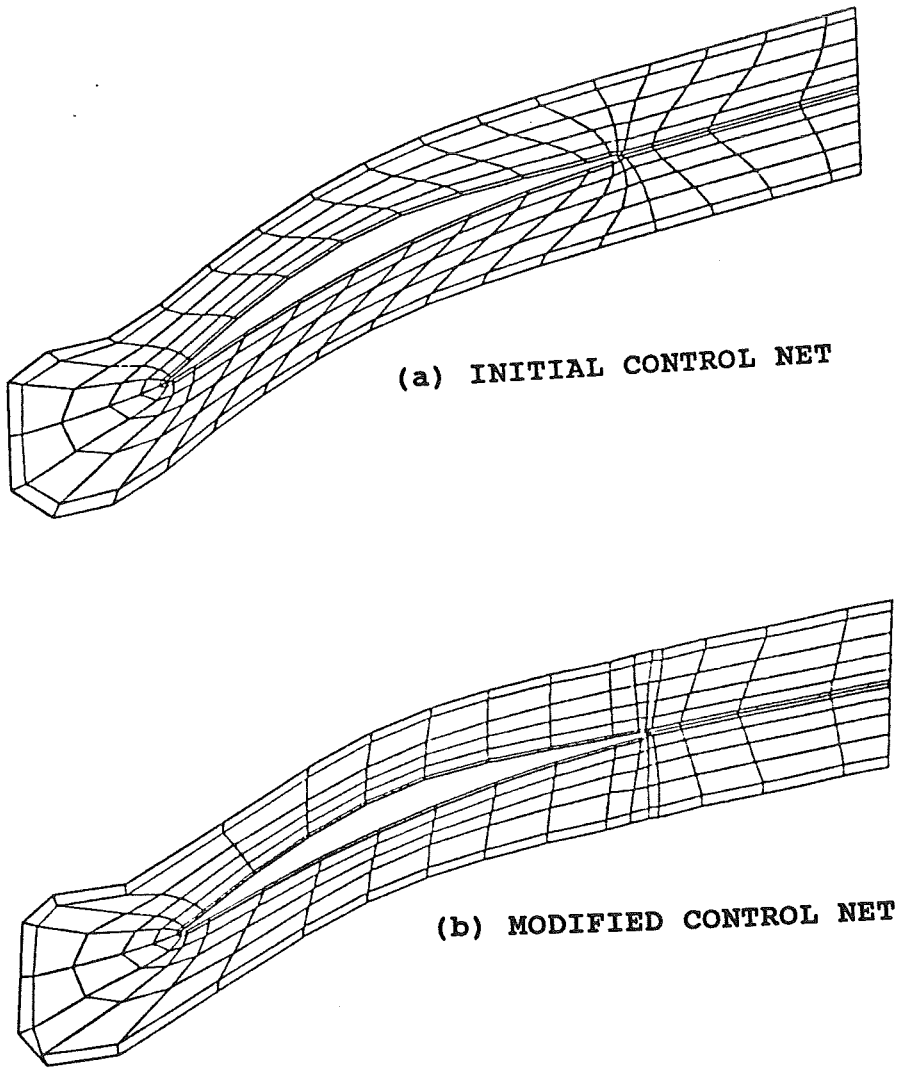
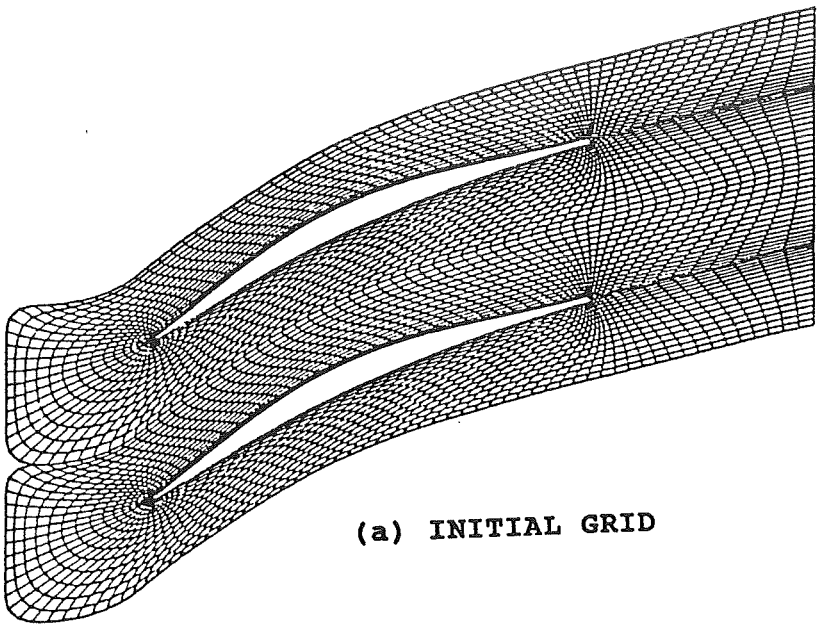
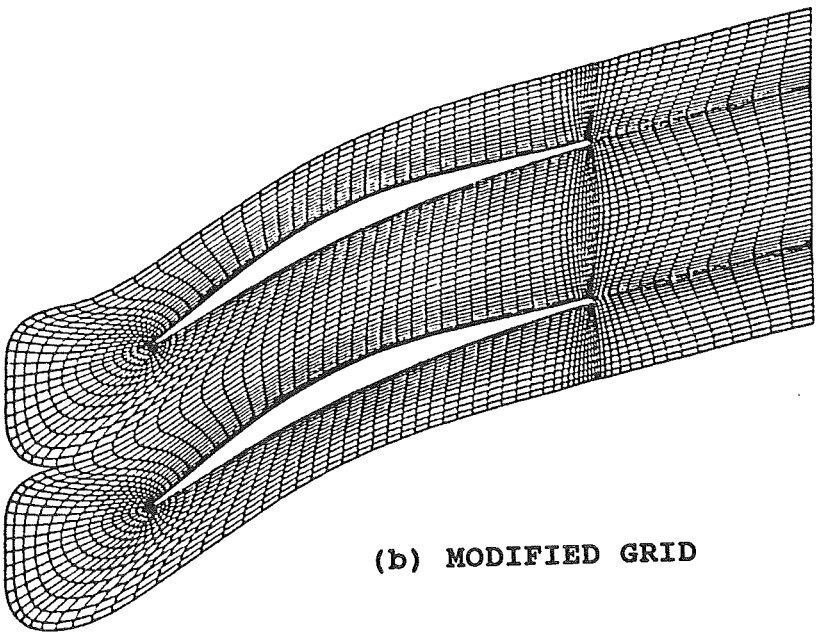


FIGURE 9. CONTROL NET FOR A COMPRESSOR ROTOR

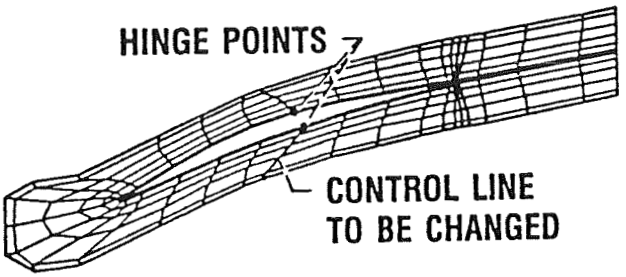


(a) INITIAL GRID

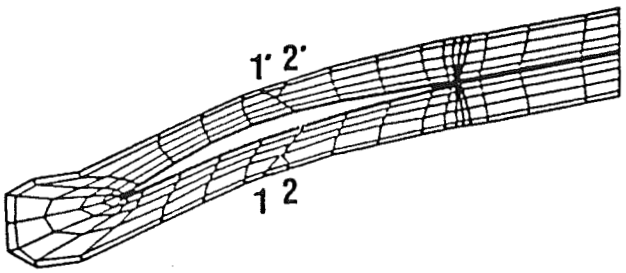


(b) MODIFIED GRID

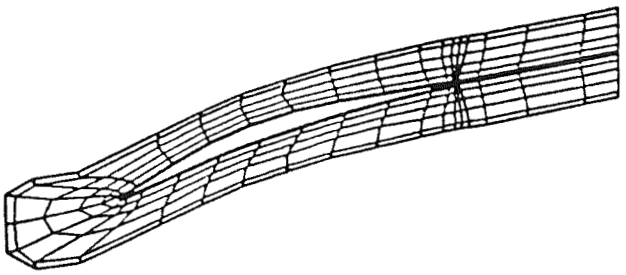
FIGURE 10. GRID COMPARISON



(a) SELECT CONTROL LINES AND HINGE POINTS



(b) TRANSLATE A CONTROL POINT FROM 1 TO 2



(c) DO "RUBBERBANDING"

FIGURE 11. A CONTROL FEATURE OF TurboT